ProtoDUNE-SP, measurements/analysis meeting
13/10/2016

Stopping Kaon Selection

Anselmo Cervera, M. Sorel

IFIC-Valencia

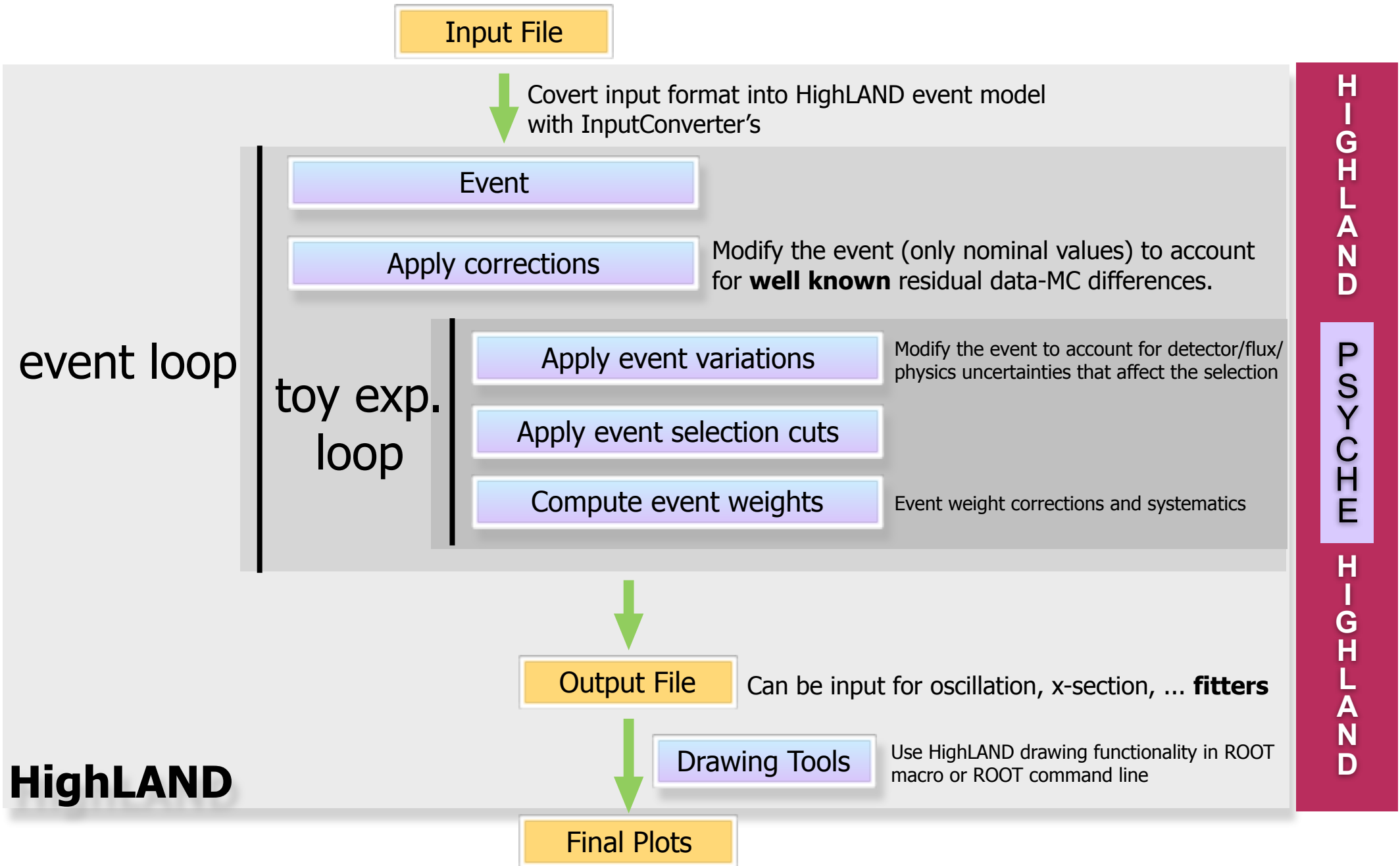
Introduction

- Look at fully simulated/reconstructed K+ events in ProtoDUNE
  - Of relevance for p → $\bar{\nu}$ K+ search in DUNE FD
  - Motivation: design event selection and optimise beam settings for maximising number of stopping kaons in ProtoDUNE
- Use HighLAND framework to do the analysis

HighLAND analysis framework

- **Highly optimized**, **thread safe, compiled c++ code** and run on the shell command line (not as root macro)
- It provides:
  - Tools for multiple simultaneous **event selections** and **systematic error propagation**
  - Tools for **drawing** the analysis results
  - **Data reduction** functionality
  - Tools for incorporating specific analyses into the framework
    - **Extensible event model** and **hierarchy of analyses**
- Presented several times at the CM and several WG meetings

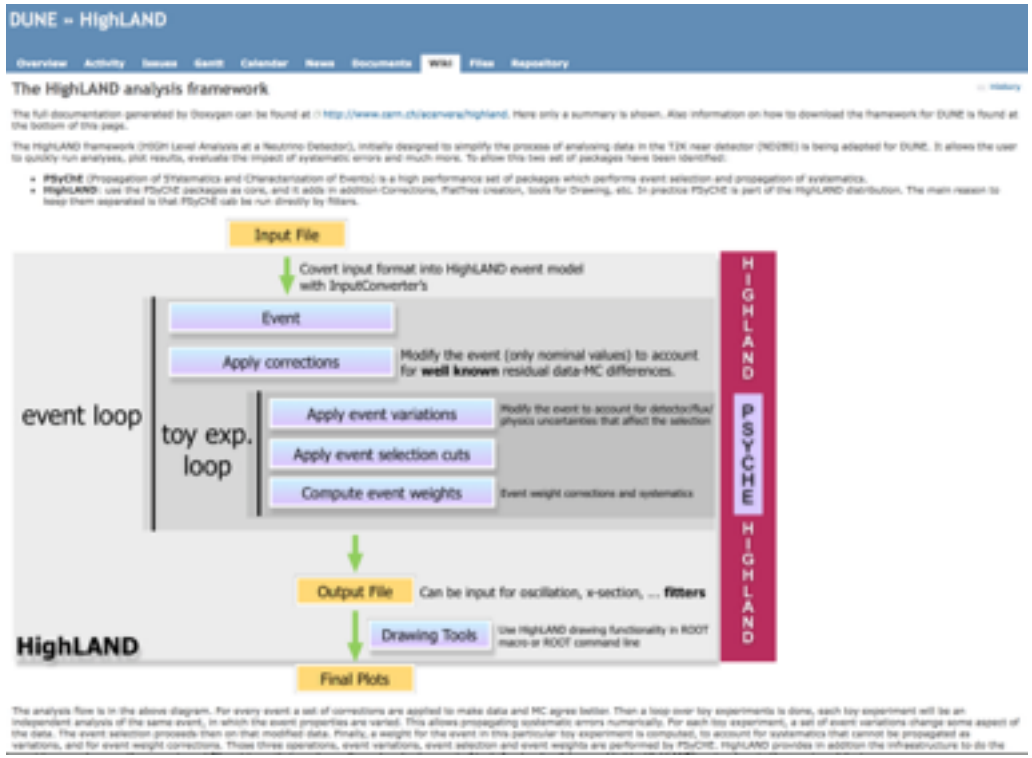
analysis flow



Documentation

- Documentation has been updated at

https://cdcvs.fnal.gov/redmine/projects/highland/wiki

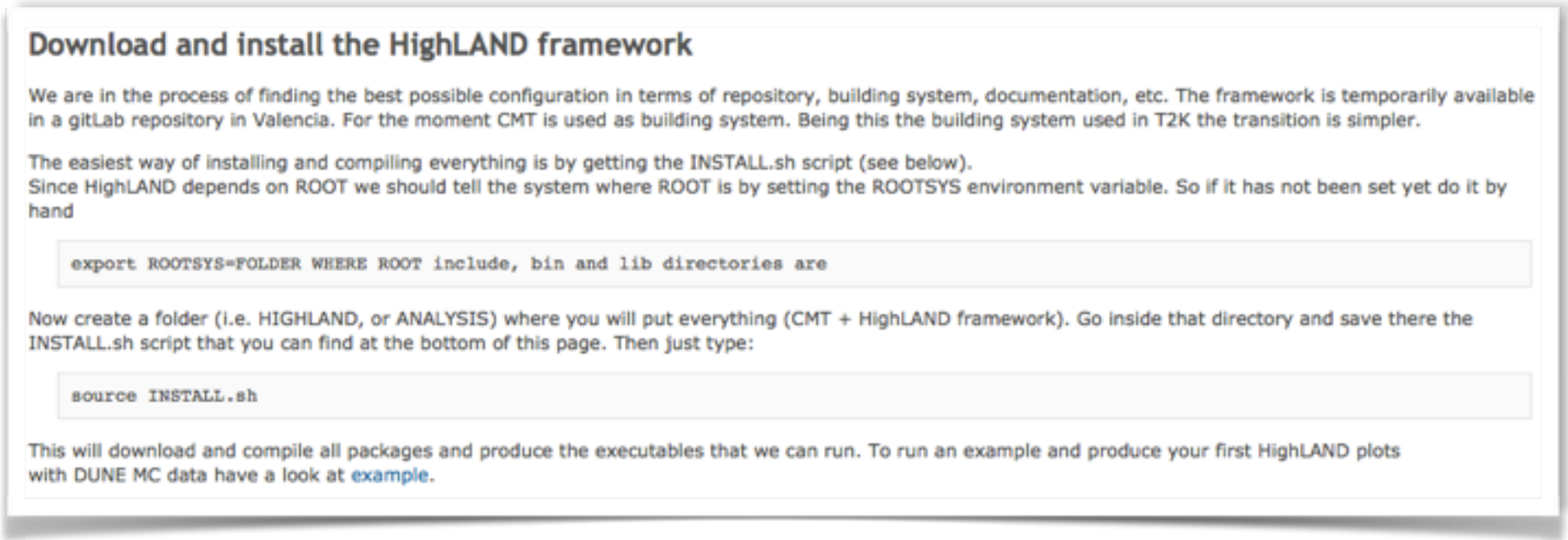


- It includes now much more information, with a link to the Doxygen documentation

Reminder: installation

- This is an screen capture of

https://cdcvs.fnal.gov/redmine/projects/highland/wiki/install



Download and install the HighLAND framework

We are in the process of finding the best possible configuration in terms of repository, building system, documentation, etc. The framework is temporarily available in a gitLab repository in Valencia. For the moment CMT is used as building system. Being this the building system used in T2K the transition is simpler.

The easiest way of installing and compiling everything is by getting the INSTALL.sh script (see below).
Since HighLAND depends on ROOT we should tell the system where ROOT is by setting the ROOTSYS environment variable. So if it has not been set yet do it by hand

```
export ROOTSYS=FOLDER WHERE ROOT include, bin and lib directories are
```

Now create a folder (i.e. HIGHLAND, or ANALYSIS) where you will put everything (CMT + HighLAND framework). Go inside that directory and save there the INSTALL.sh script that you can find at the bottom of this page. Then just type:

```
source INSTALL.sh
```

This will download and compile all packages and produce the executables that we can run. To run an example and produce your first HighLAND plots with DUNE MC data have a look at example.

- Please try, and if you have any problems email me (acervera@ific.uv.es) or submit an issue to **redmine**

https://cdcvs.fnal.gov/redmine/projects/highland/issues

ProtoDUNE example

- Highland could be specially useful for ProtoDUNE
  - Given the time constraints
  - Suitable for both prototypes
    - The same analysis could be performed in both prototypes, facilitating the comparisons
- An example have been committed to the git repository: **protoDuneAnalysisExample** package:
  - stoppingKaonSelection
  - dEdxCorrection, dEdxVariation (systematic)
- The new selection is being used for proton decay related studies and to optimize the beam momentum for kaons

See Michel Sorel NDK talk at last CM

https://indico.fnal.gov/getFile.py/access?contribId=129&sessionId=10&resId=0&materialId=slides&confId=10613

Event Selection

- A selection is a collection of steps (cuts or actions)
- Each selection inherits from **SelectionBase**, which has a main mandatory method **DefineSteps**

```
//**********************************************************************
void stoppingKaonSelection::DefineSteps(){
//**********************************************************************

  // Steps must be added in the right order
  // if "true" is added to the constructor of the step,
  // the step sequence is broken if cut is not passed (default is "false")
  AddStep(StepBase::kCut,    "> 0 tracks",         new AtLeastOneTrackCut());
  AddStep(StepBase::kAction, "find true vertex",   new FindTrueVertexAction_proto());
  AddStep(StepBase::kAction, "find main track",    new FindMainTrackAction());
  AddStep(StepBase::kAction, "find vertex",        new FindVertexAction());  // action from duneExampleAnalysis package
  AddStep(StepBase::kCut,    "kaon range",         new KaonRangeCut());
  AddStep(StepBase::kCut,    "> 1 track",          new MoreThanOneTrackCut());
  AddStep(StepBase::kCut,    "PIDA",               new PIDACut());
```

stoppingKaonSelection

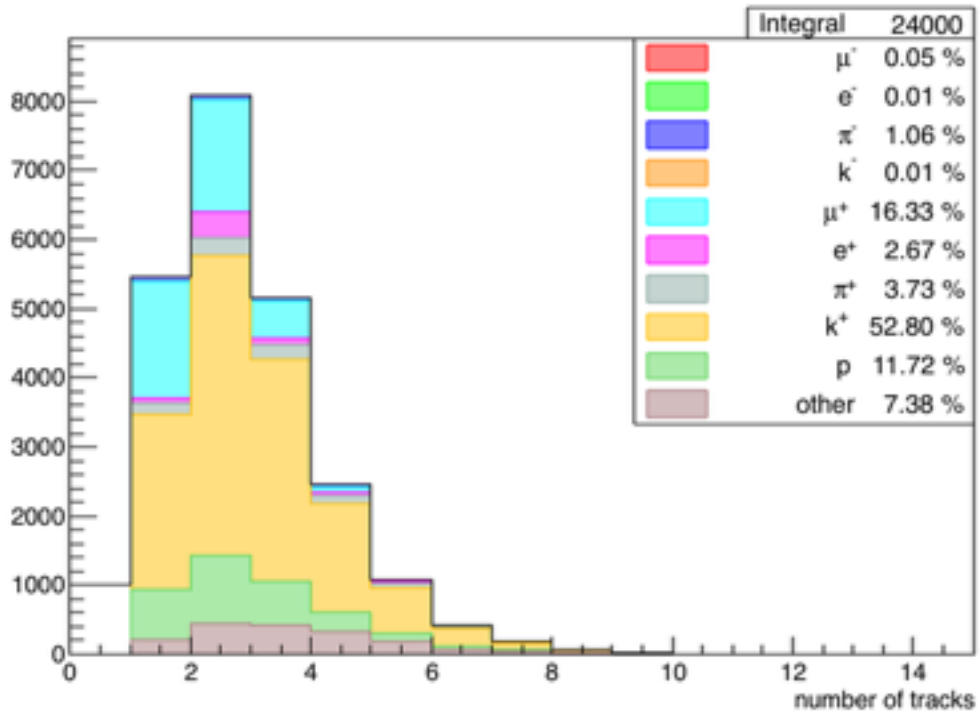
- Each step inherits from **StepBase** and implements the method **Apply**

```
//**************************************************
bool KaonRangeCut::Apply(AnaEventC& event, ToyBoxB& boxB) const{
//**************************************************

  // Cast the ToyBox to the appropriate type
  ToyBoxDUNE& box = *static_cast<ToyBoxDUNE*>(&boxB);
  if (!box.MainTrack) return false;

  Float_t length = static_cast<AnaParticle*>(box.MainTrack)->Length;
  if (fabs(length-200)<10) return true;
  else return false;
}
```

Results for 1 GeV/c kaons from MCC7

Track multiplicity

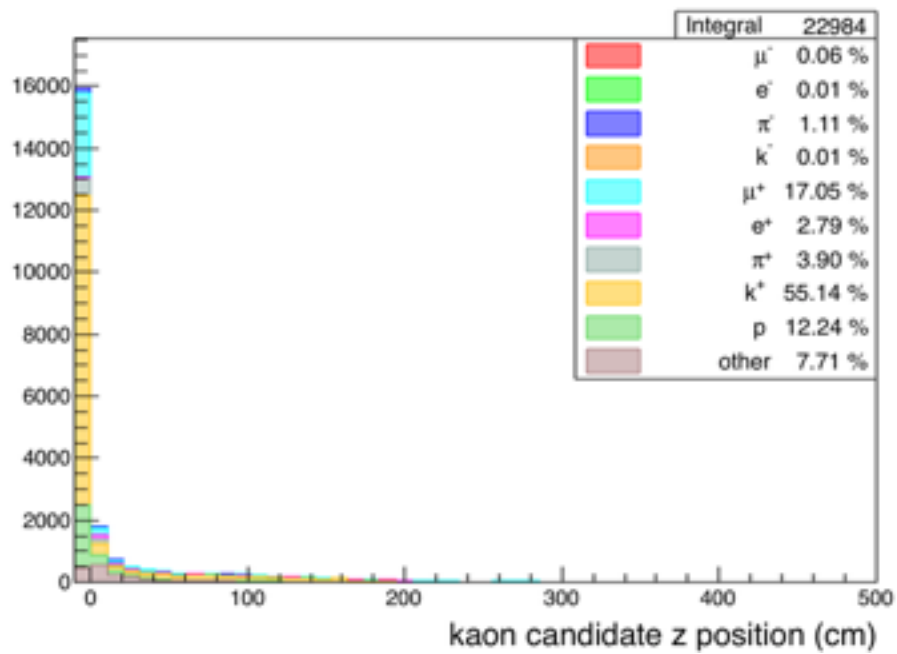
- About 95% of events have at least one reconstructed track
- 1-4 reconstructed tracks is typical
- When >0 tracks, color indicates true particle associated with kaon candidate track in the event (see next)

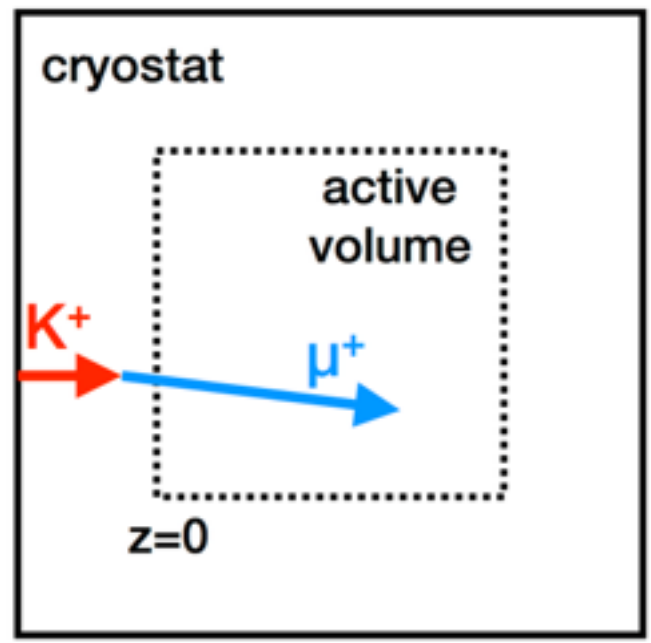



```
draw.SetTitleX("number of tracks");
draw.Draw(default,"ntracks",15,0,15,"particle","accum_level>-1","HIST","DRAWALLMC PUR");
```

Kaon candidate track

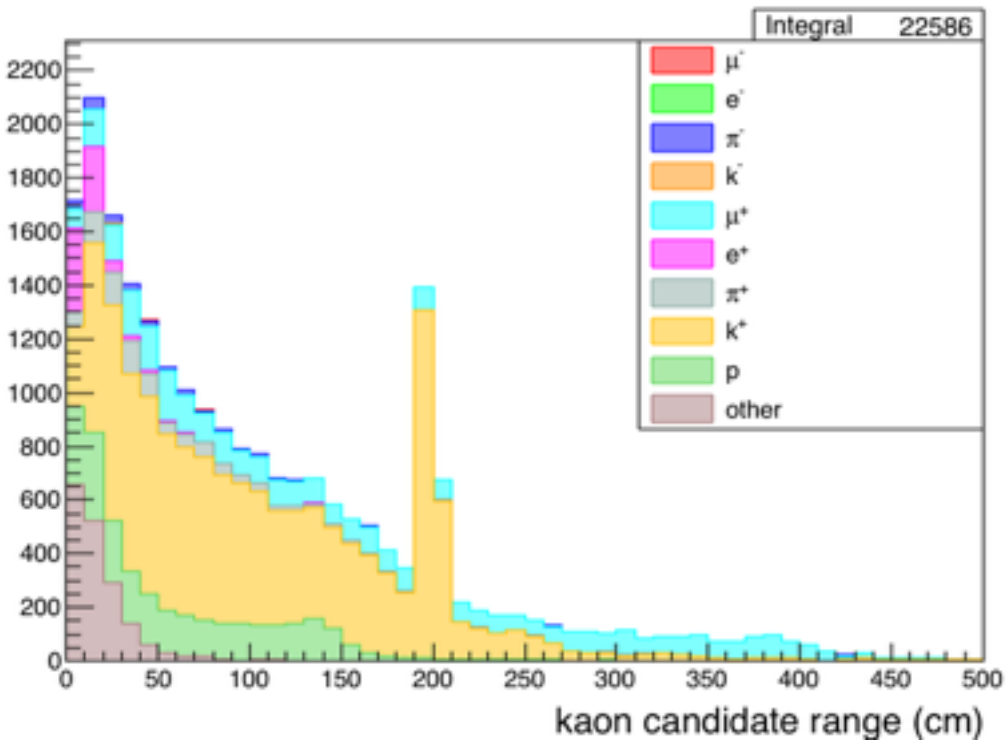
- Kaon candidate track defined as track starting most upstream (lowest z)
- With this definition, kaon candidate associated to true kaon only ~50% of the times
  - Partly because kaon decays or interacts before reaching the active volume (z>0) in ~30% of the simulated events

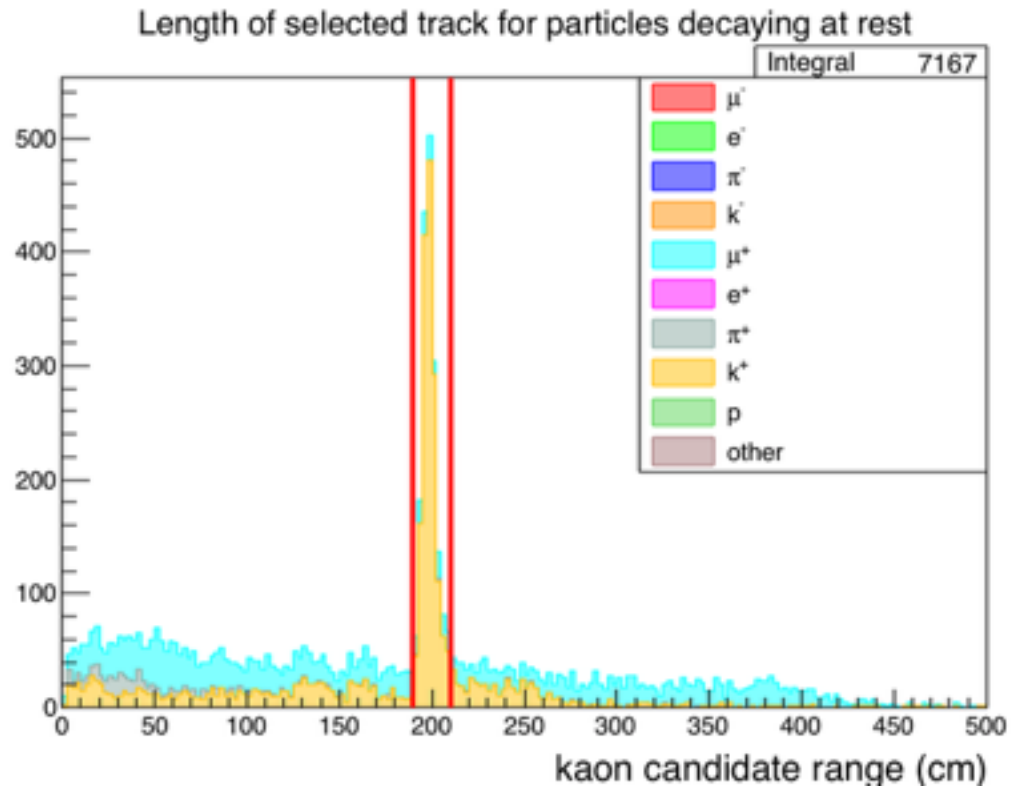




Fraction of stopping kaons and range cut

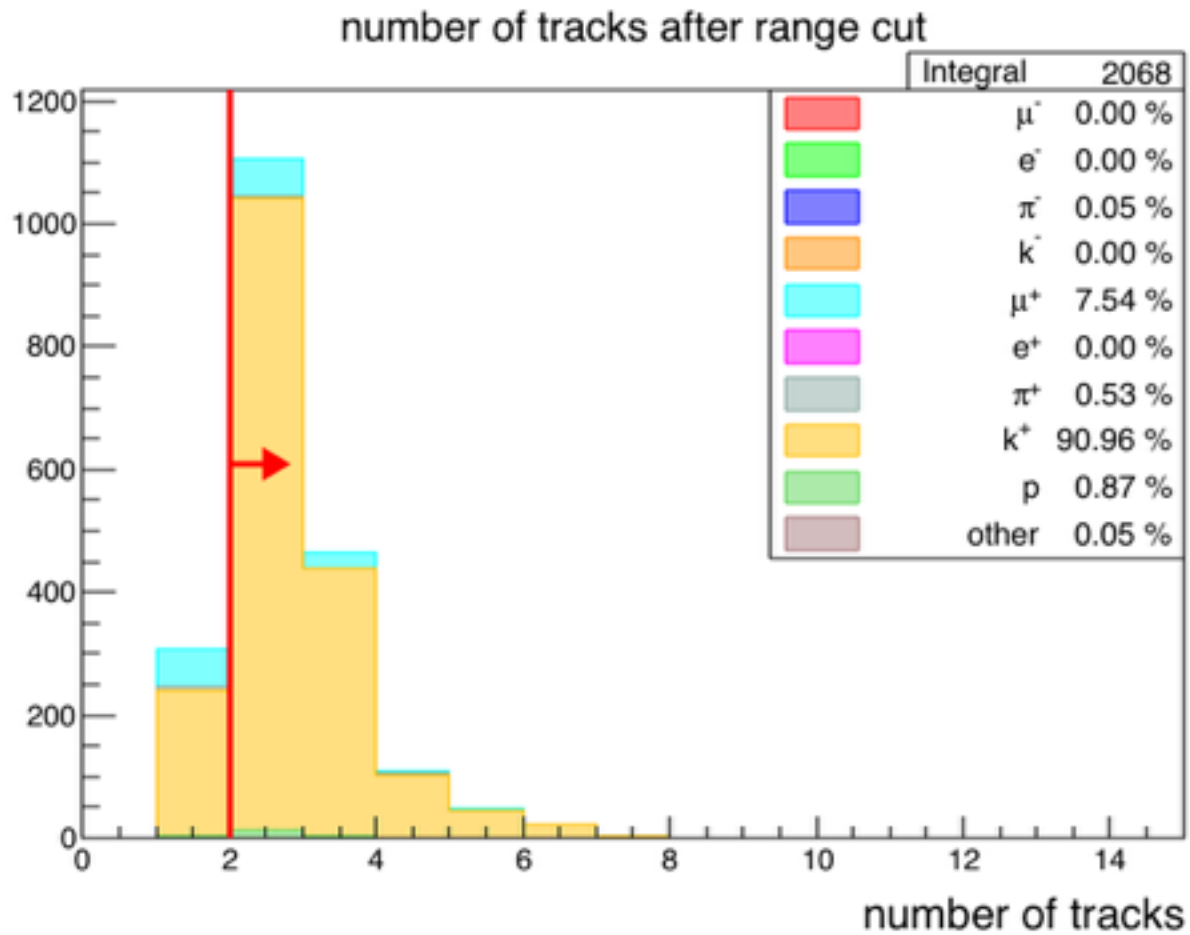
- Only ~10% of the events have a kaon stopping in the active volume
- Require ~200 cm range tracks to select all correctly reconstructed kaons decaying at rest





Multiplicity cut

- In addition to track range, >1 track requirement further improves kaon decay at rest selection
- Reason: tracks from kaon daughters are expected

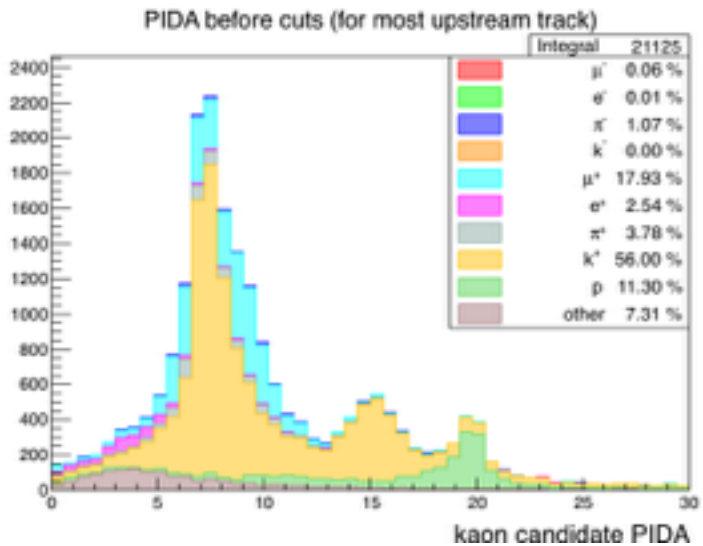


The PIDA variable

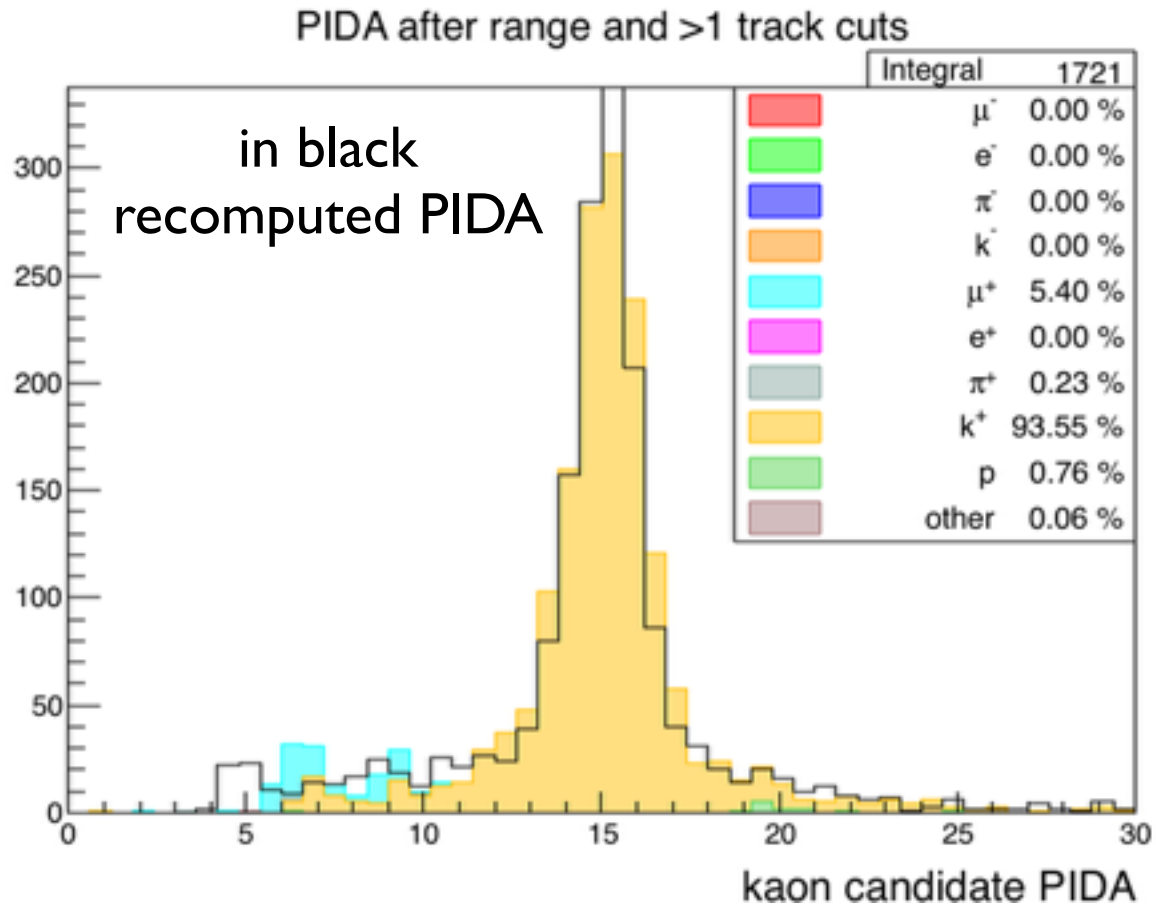
- Averaged over all hits with residual range R<30

$$PIDA =< A_i >=< (dE/dx)_i R_i^{0.42} >$$

before cuts

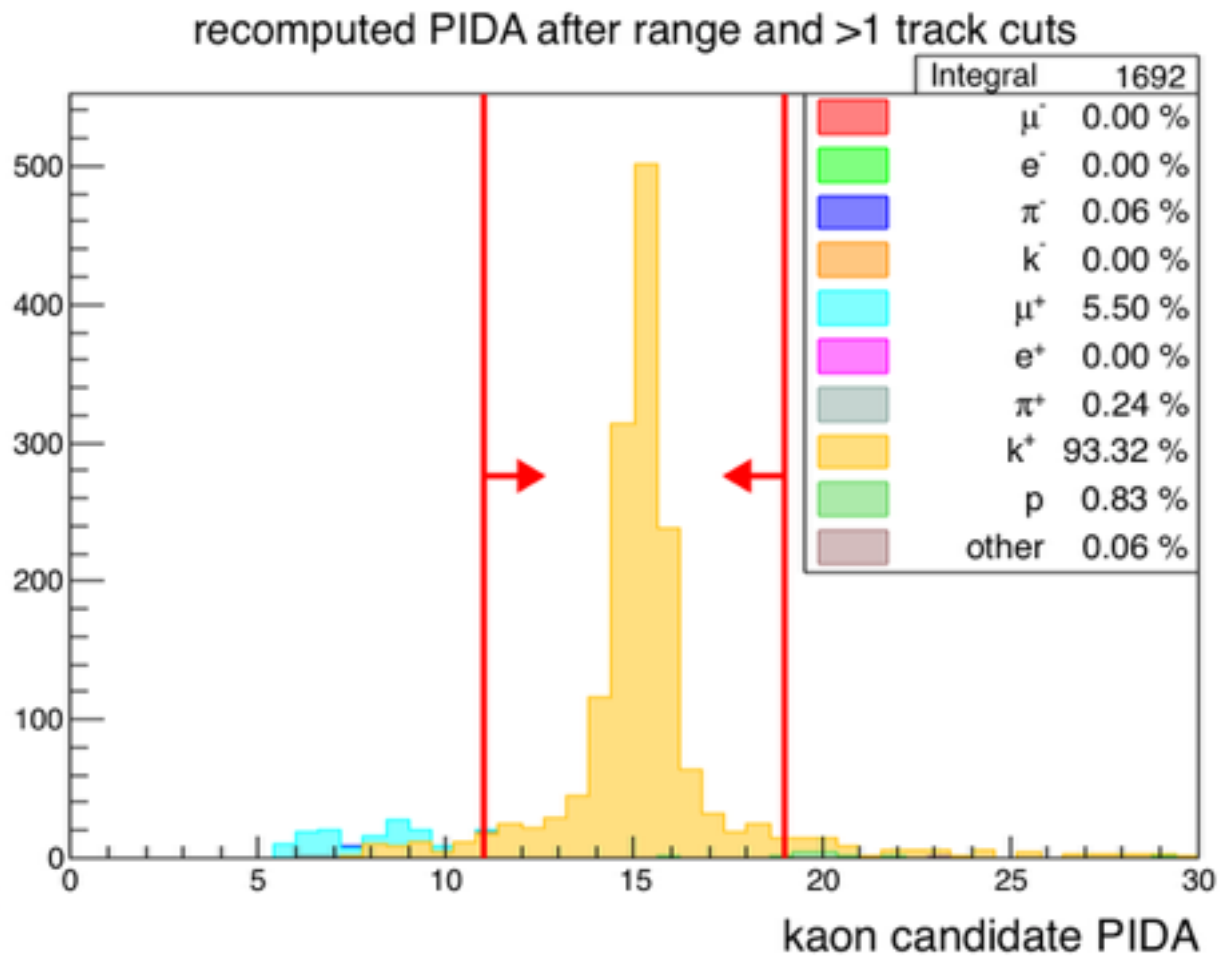



```
//********************************************************************
Float_t anaUtils::ComputePIDA(const AnaParticleB &track) {
//********************************************************************

  Float_t cut=30;

  Float_t PIDA=0;
  Int_t ncontrib=0;
  for (Int_t i=0;i<3;i++){
    for (Int_t j=0;j<track.NHitsPerPlane[i];j++){
      if (track.ResidualRange[i][j]<cut && track.ResidualRange[i][j]>0){
        ncontrib++;
        PIDA += track.dEdx[i][j]*pow(track.ResidualRange[i][j],0.42);
      }
    }
  }
  if (ncontrib>0) PIDA /= ncontrib*1.;

  return PIDA;
}
```



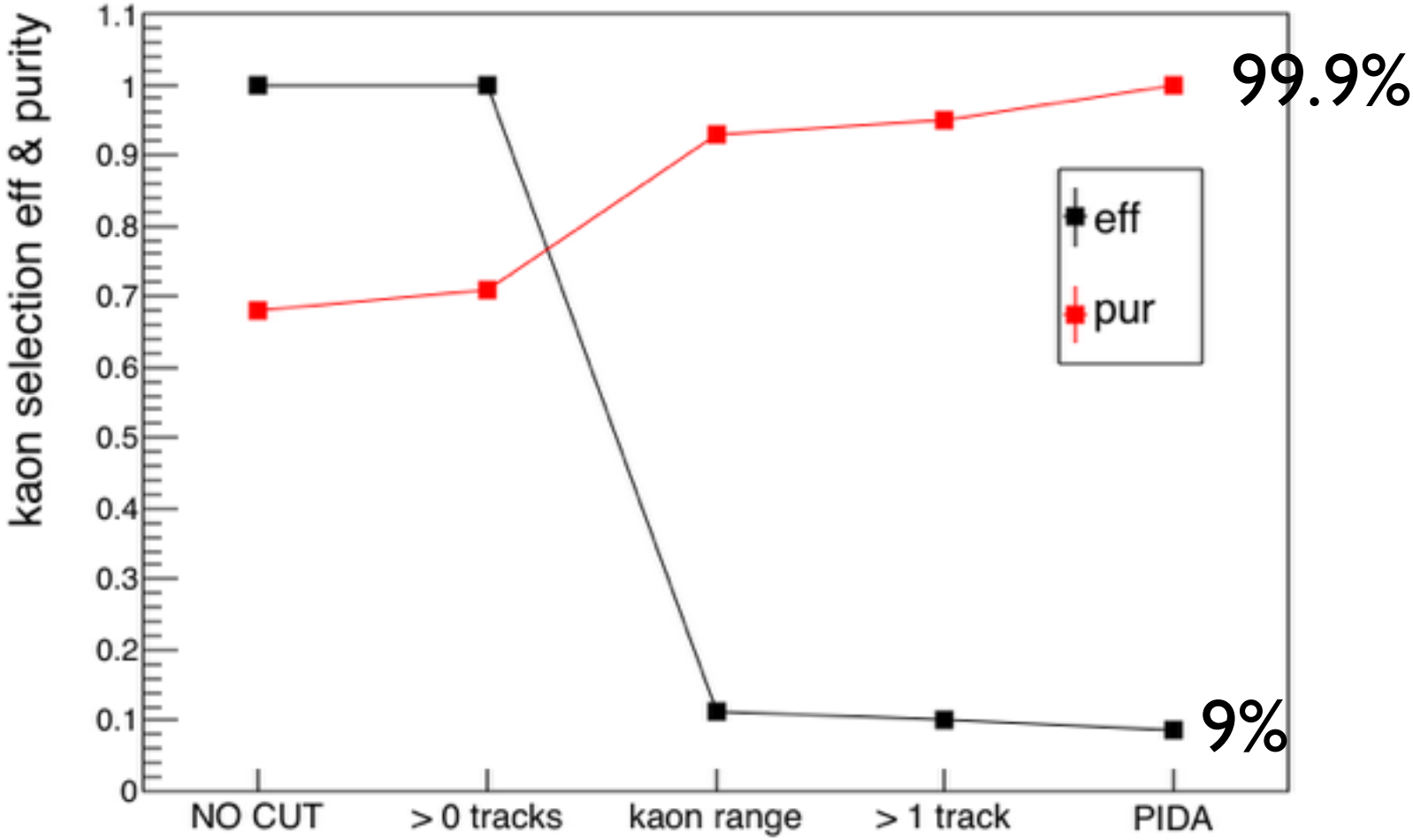
The recomputed PIDA is narrower, why ? We think
it is because we have used all 3 wire planes, while the one in the AnaTree only use one

The PIDA cut




```
//**************************************************
bool PIDACut::Apply(AnaEventC& event, ToyBoxB& boxB) const{
//**************************************************

  (void)event;

  // Cast the ToyBox to the appropriate type
  ToyBoxDUNE& box = *static_cast<ToyBoxDUNE*>(&boxB);
  if (!box.MainTrack) return false;

  Float_t pida = anaUtils::ComputePIDA(*box.MainTrack);
  if (fabs(pida-15.)<4) return true;
  else return false;
}
```

Efficiency and purity




```
// Create a data sample instance with the micro-tree file.
// Needed to plot efficiency (from truth tree) and purity (from default tree) simultaneously
DataSample mc("test.root");

// kaon selection Efficiency and purity after each cut
draw.SetTitleY("kaon selection eff & purity");
draw.DrawEffPurVSCut(mc,"true_signal==1");
```

Other kaon momenta

1.5, 2, and 3 GeV/c

Samples

- Samples produced by Elizabeth (thanks !!!):
  - 1.5, 2 and 3 GeV/c
  - 1000 events each
- Motivation:
  - Found optimal beam setting since we know there are no kaons arriving to the detector at 1GeV/c, while there are at 2 GeV/c and it does not increase much at 3 GeV/c

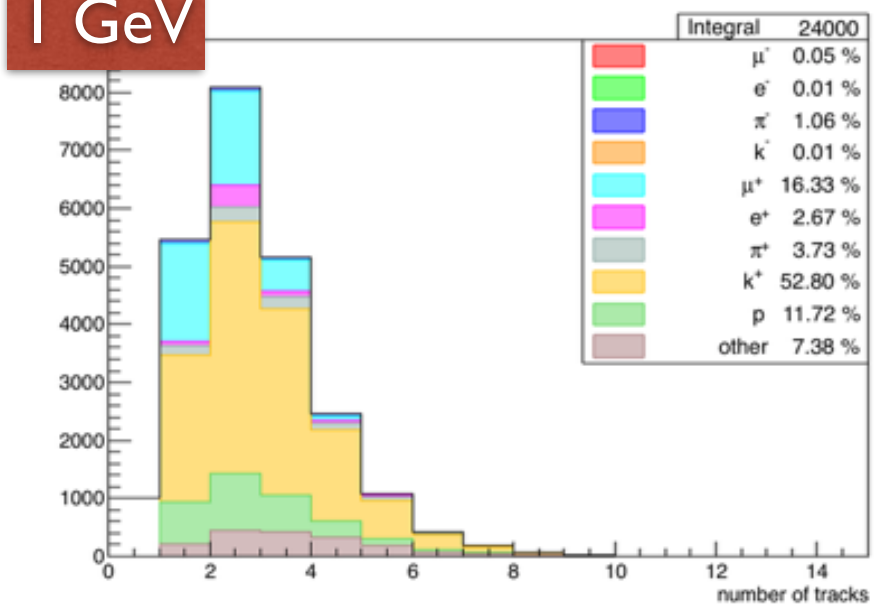
Positive Hadron Beam:

P (GeV/c)	# Spills	e^+	K^+	μ^+	p	π^+	# Triggers	Beam time (days)
1	20K	672K	~0	3K	192K	144K	1M	5.6
2	20K	480K	8K	21K	336K	480K	1.3M	5.6
3	20K	760K	8K	8K	101K	321K	1.2M	5.6
4	20K	559K	25K	16K	99K	501K	1.2M	5.6
5	20K	448K	32K	8K	104K	608K	1.2M	5.6
6	20K	334K	50K	7K	121K	689K	1.2M	5.6
7	20K	251K	55K	12K	129K	753K	1.2M	5.6

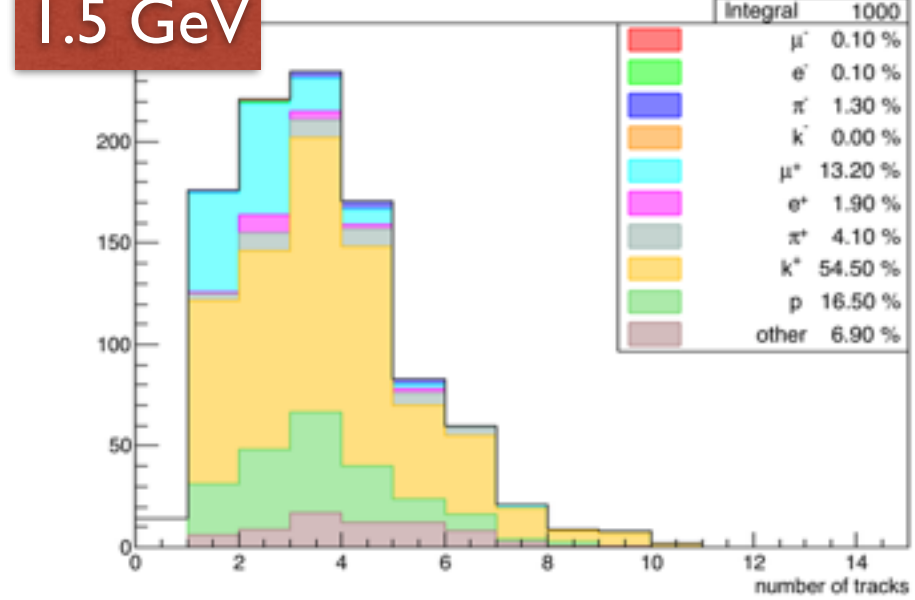
https://indico.fnal.gov/getFile.py/access?contribId=40&sessionId=10&resId=0&materialId=slides&confId=10613

Track multiplicity

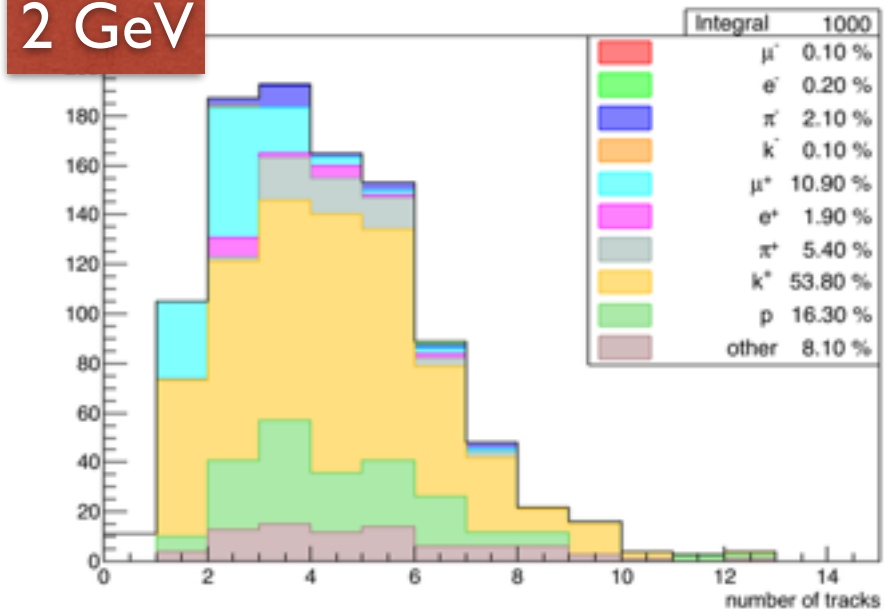
1 GeV



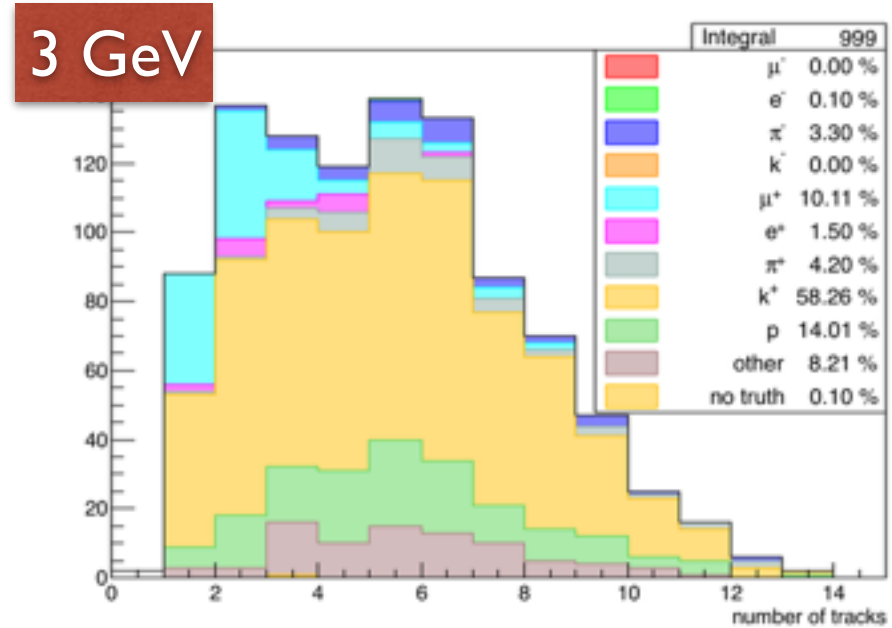
1.5 GeV



2 GeV

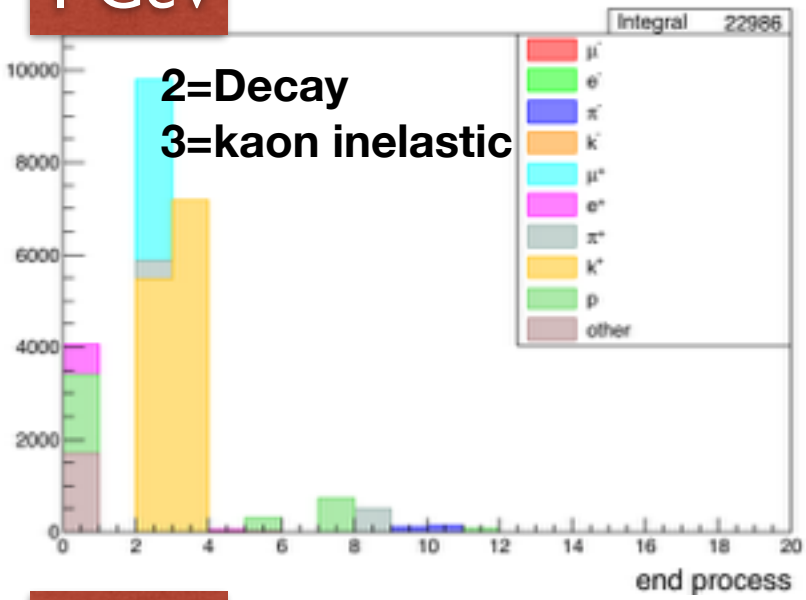


3 GeV

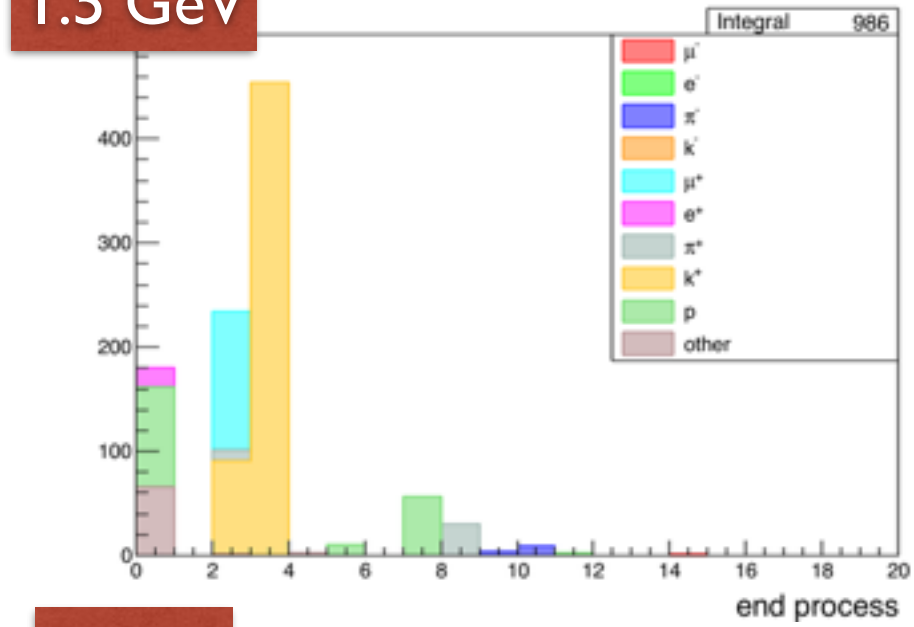


Fraction of kaons decaying

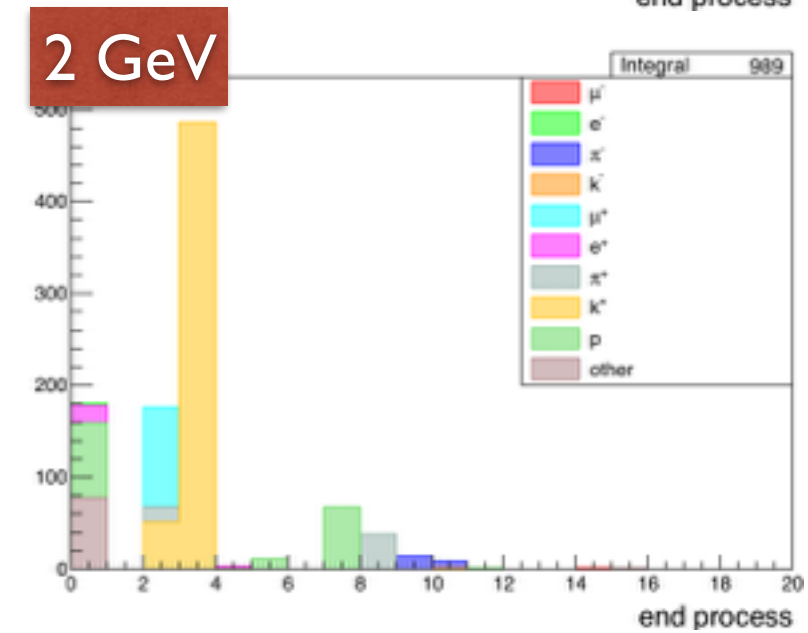
1 GeV



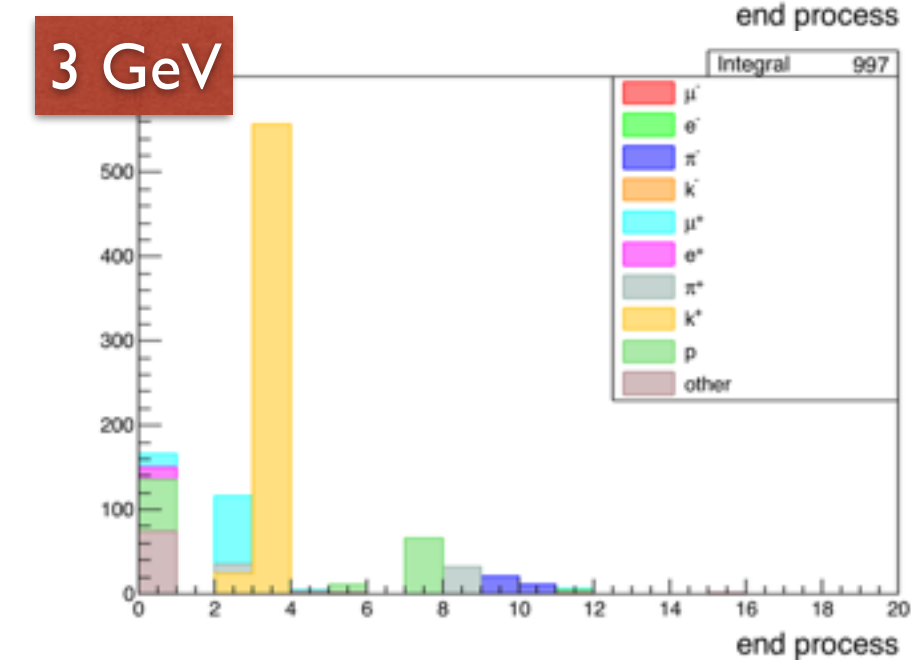
1.5 GeV



2 GeV

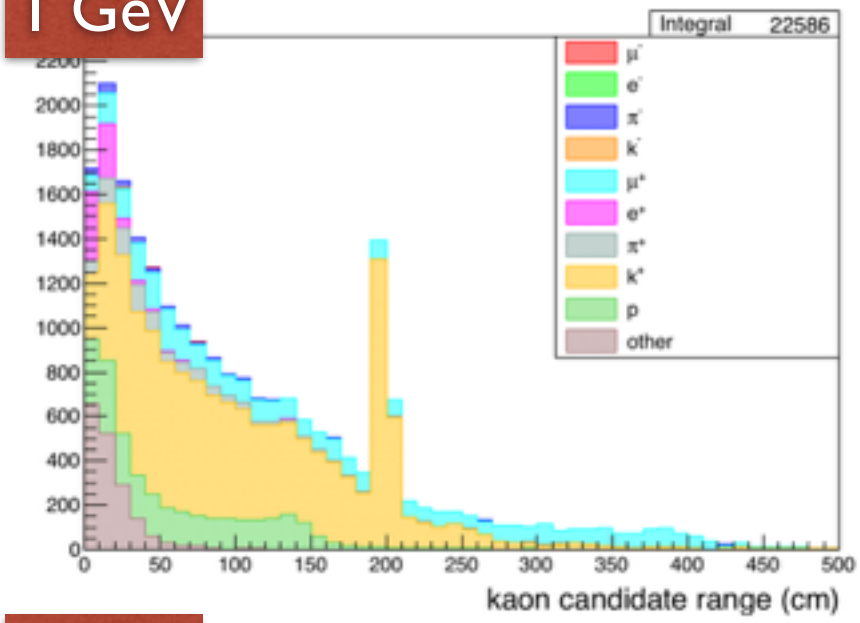


3 GeV

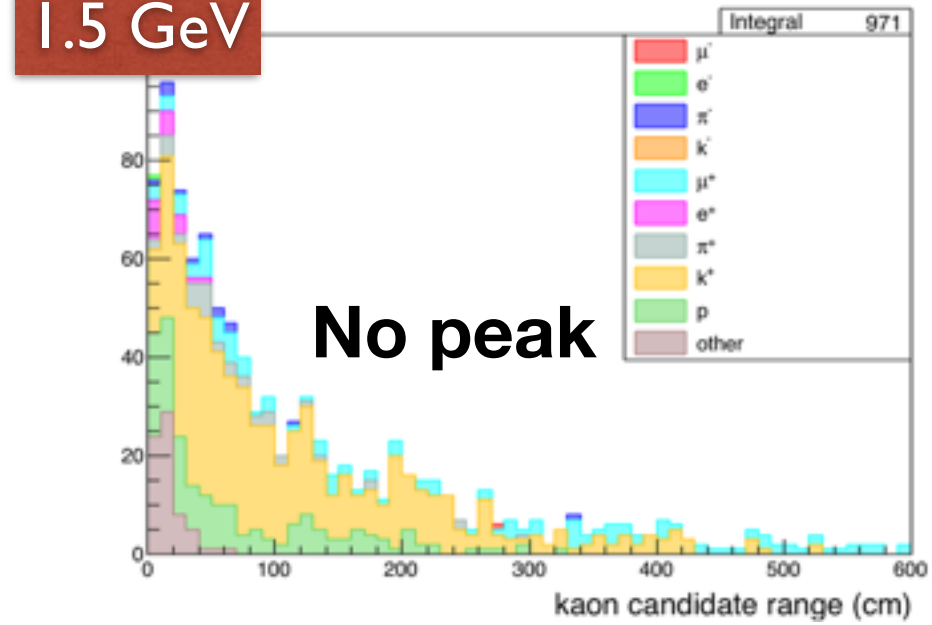


Kaon kandidate Range

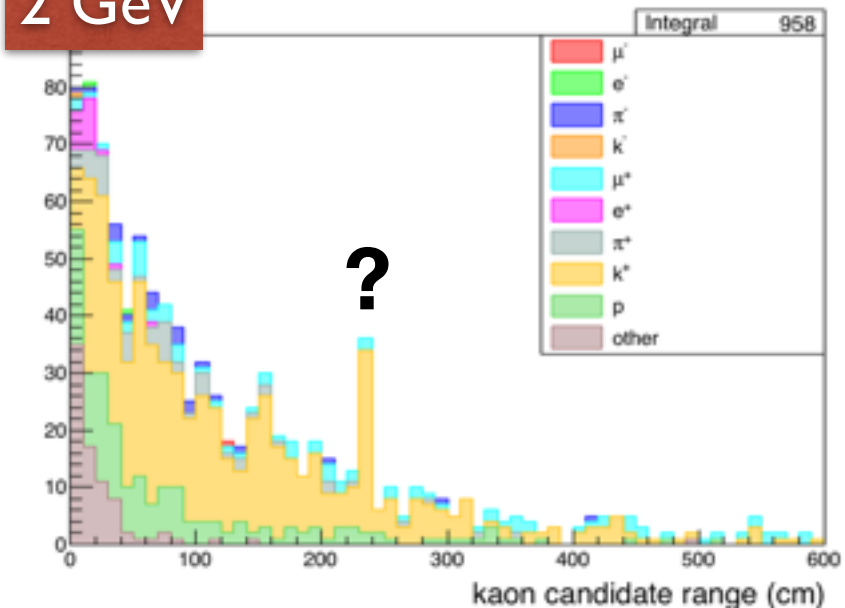
1 GeV



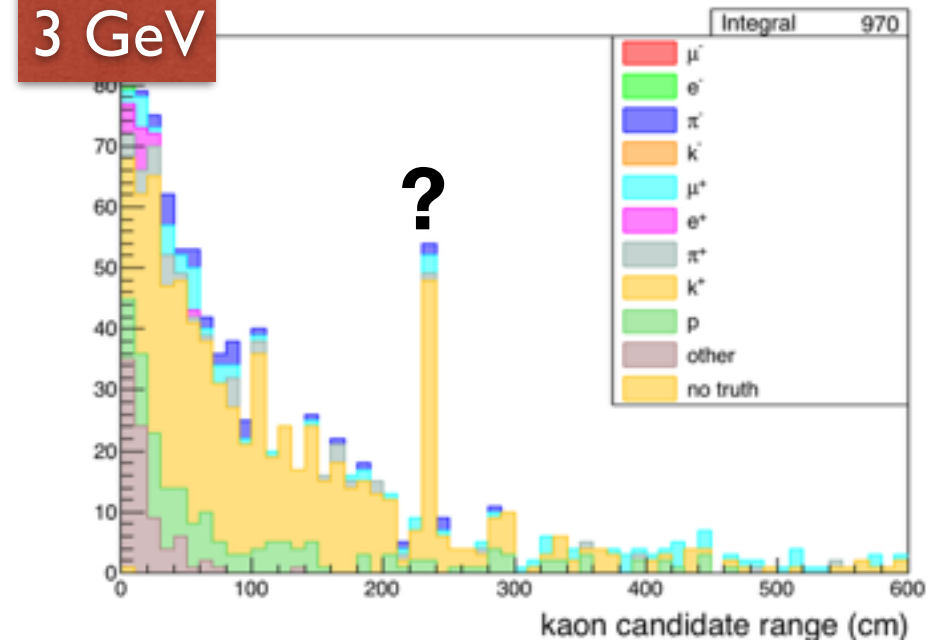
1.5 GeV



2 GeV

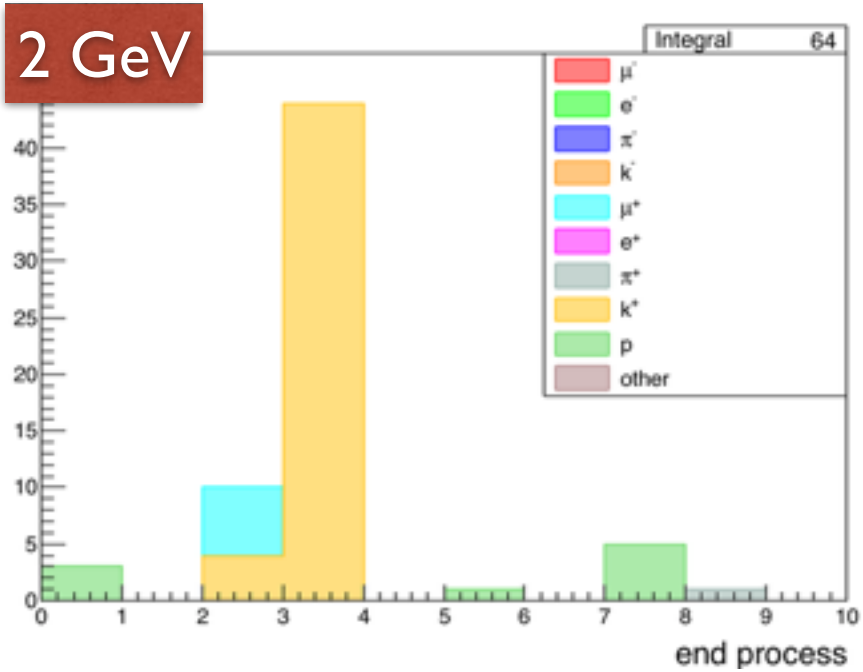


3 GeV

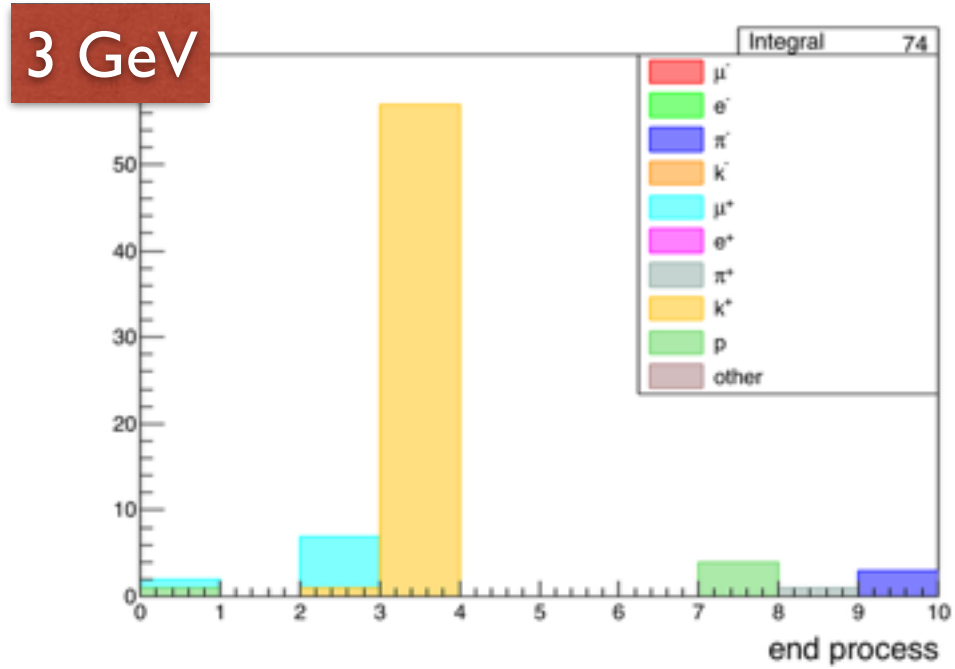


- It seems that 2 and 3 GeV are decaying at rest at a fixed range, but the end process tells us that the kaon undergoes an inelastic interaction
- Plots for |range-235|<20 (peaks in previous slide)
- Could it be interactions in the cathode ?

2 GeV

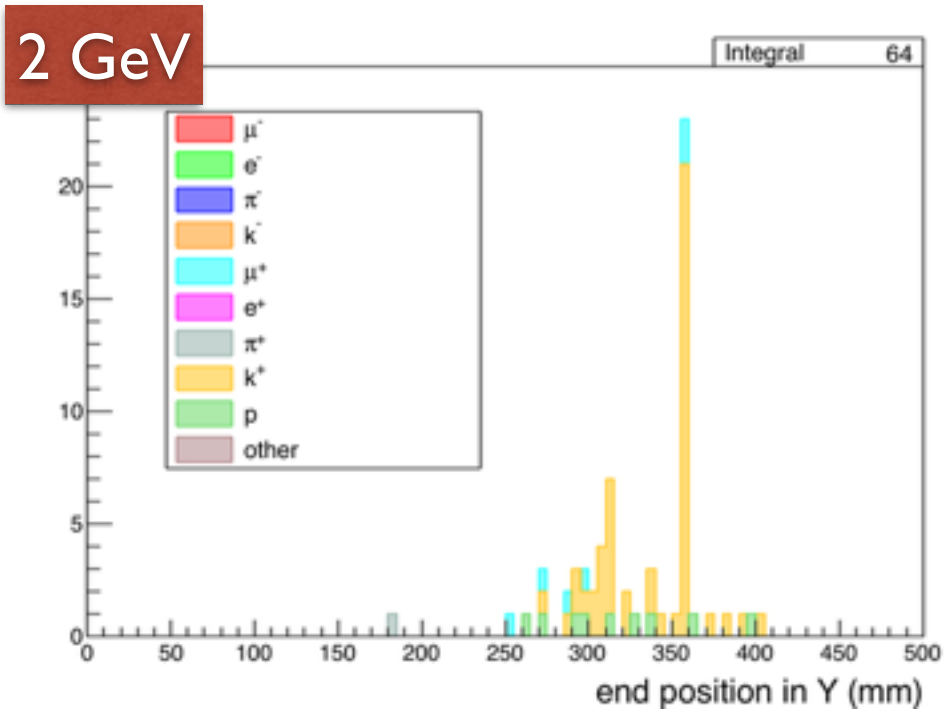


3 GeV

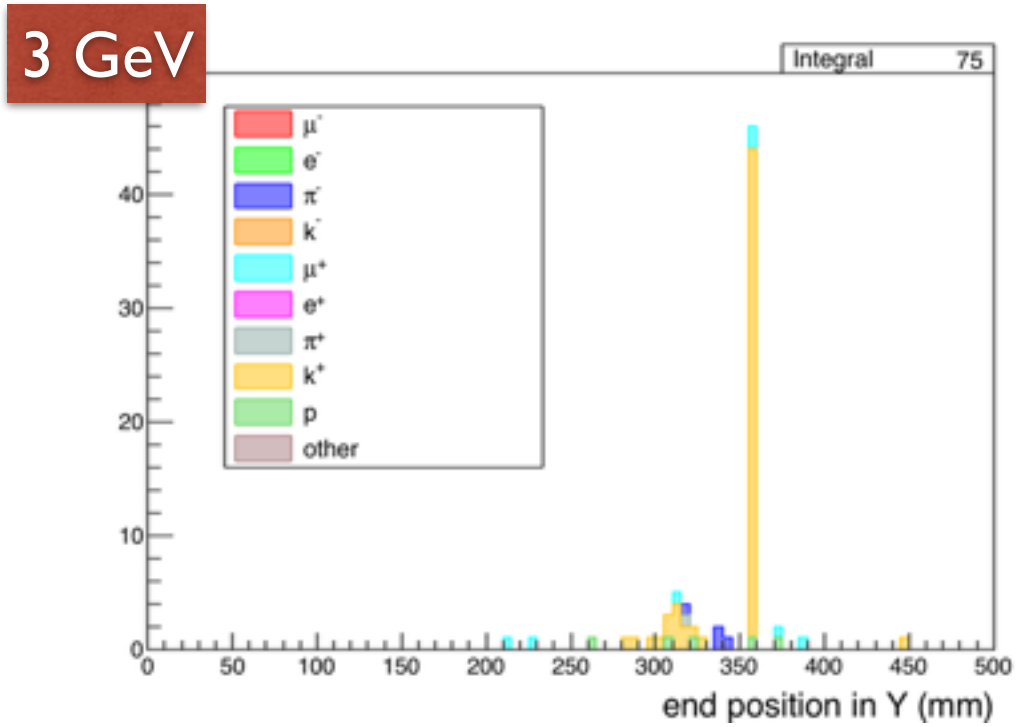


- Plots for |range-235|<20
- I guess the peak at 360 mm corresponds to the cathode

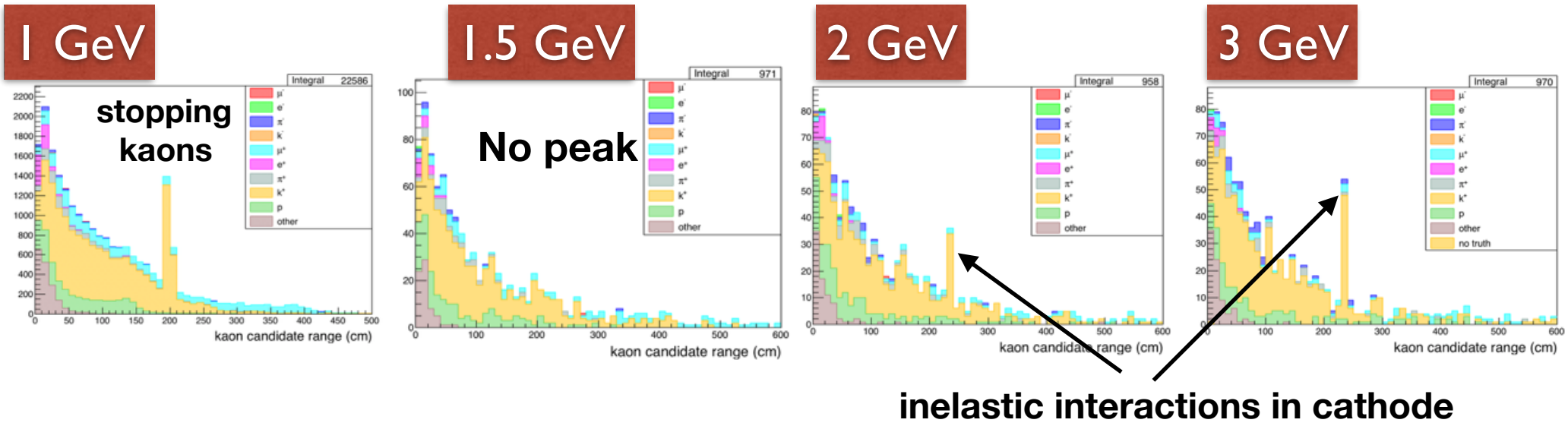
2 GeV



3 GeV



Summary



- It seems that there are no kaons decaying at rest above 1.5 GeV/c
  - Is this what we expect from first principles ? Notice that from 1 to 1.5 GeV makes a difference for a particle with 0.5 GeV mass
- Should we look at lower energies: between 1 and 1.5 GeV/c?
- Do we expect a significant amount of kaons arriving to the detector at those energies ?

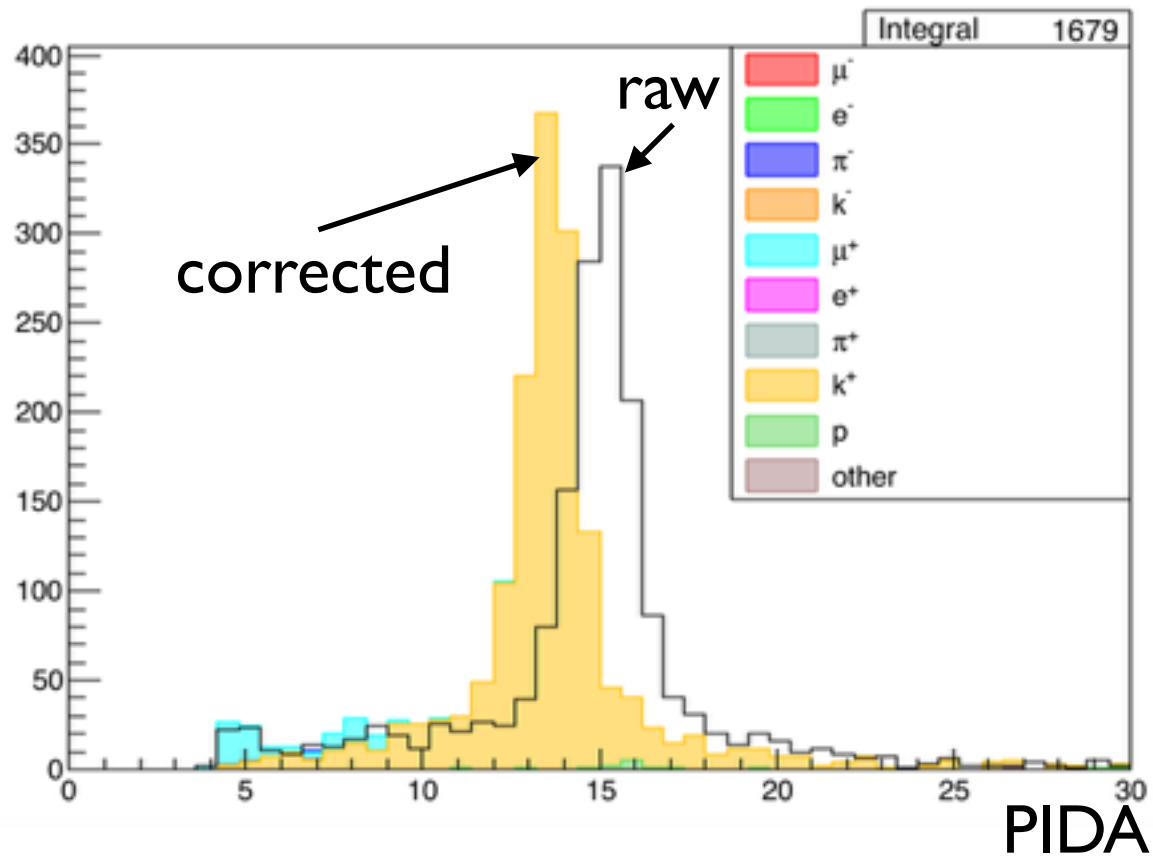
Additional studies with highland

Corrections

- Correct a well known data MC difference to reduce the corresponding systematic
- Example: **dEdxCorrection**
  - Scales the dEdx of each hit by the correction factor and recomputes PIDA

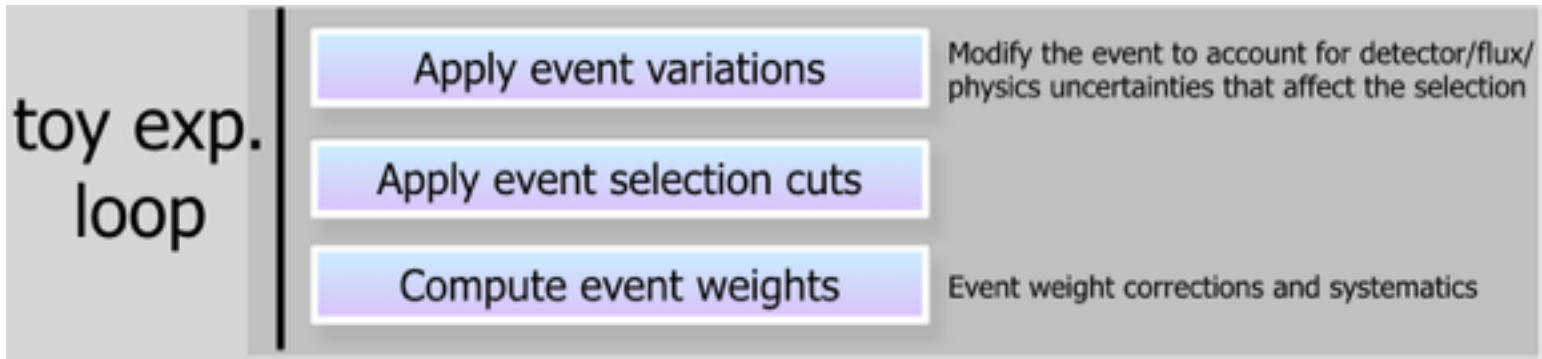
data/dEdx.dat

bins of PDG		correction	error on correction
10	12	0.95	0.02
12	14	0.98	0.02
320	322	0.9	0.02
2211	2213	0.8	0.02
210	212	0.9	0.02



Systematics

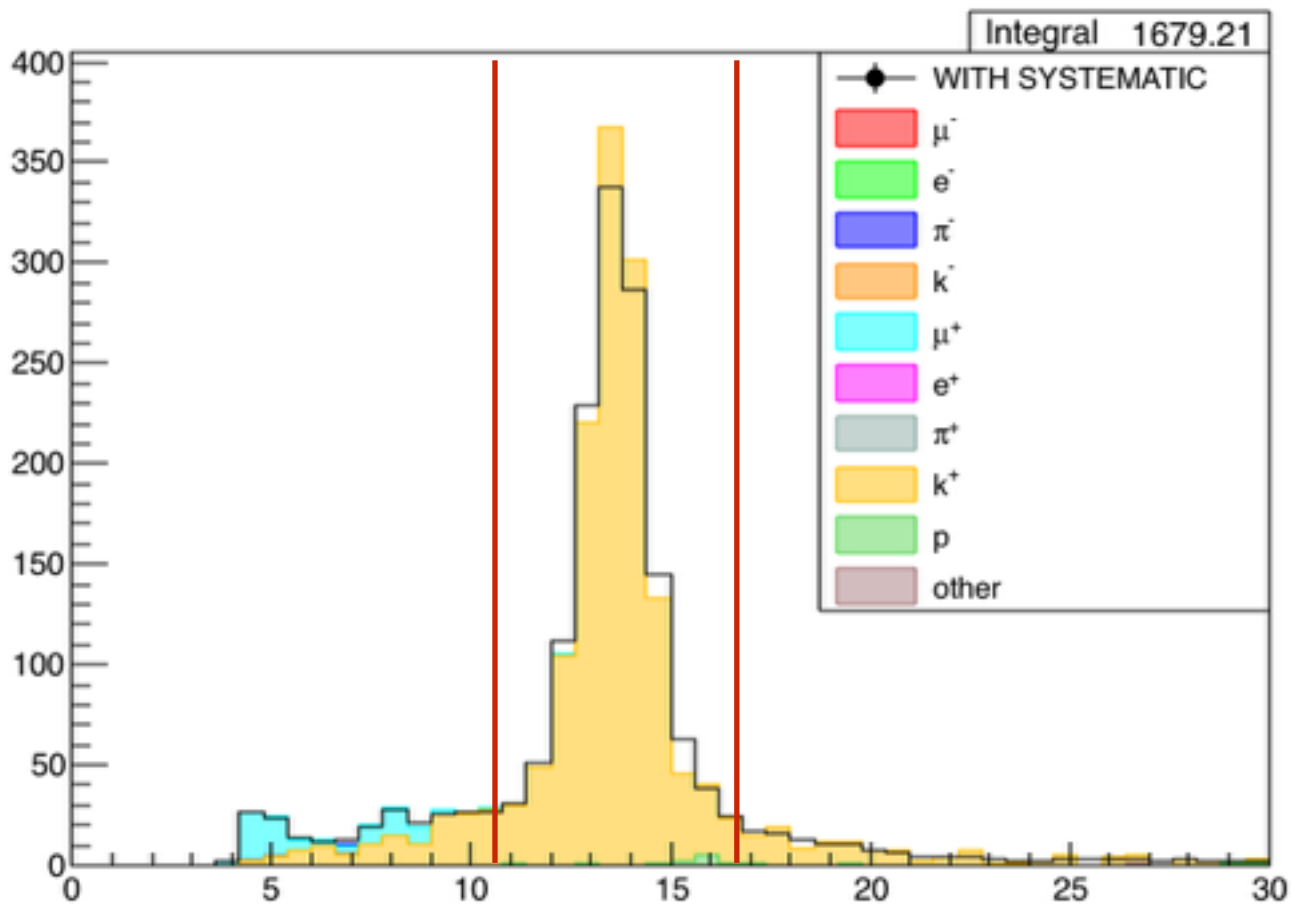
- full systematic propagation functionality is one of the main HighLAND benefits
- Systematic are propagated numerically by multiple throws (toy experiments)
- Two type of systematics:
  - **Event Variations**: Modify the input data
  - **Event Weights**: Just a global weight for the event



dEdxVariation systematic

- The error on the correction is the systematic
- 100 toy experiments. Each toy applies a different correction factor to all hits. Then PIDA is recalculated

bins of PDG		correction	error on correction
10	12	0.95	0.02
12	14	0.98	0.02
320	322	0.9	0.02
2211	2213	0.8	0.02
210	212	0.9	0.02



Effect on the selection

- When the PIDA cut is applied the dEdx systematic has an effect on the number of selected events
  - Integrated: **0.3%**
  - Differential: **< 3%**

for events passing all cuts, including PIDA